Serwery XMPP

Mateusz Kaźmierczak
Karol Czosnowski

1. Czym jest XMPP?

XMPP (Extensible Messaging and Presence Protocol) jest protokołem bazującym na języku XML umożliwiającym przesyłanie w czasie rzeczywistym wiadomości, statusu oraz usług typu żądanie-odpowiedź. Protokół ma zastosowanie nie tylko w komunikatorach, ale również w innych systemach natychmiastowej wymiany informacji. Mimo iż nazwa konta na serwerach XMPP jest skonstruowana podobnie jak adres e-mail, nie może być on wykorzystywany jako ten sposób komunikacji. Dzieje się tak gdyż ten protokół zakłada wyłącznie błyskawiczną formę komunikacji między użytkownikami.

Początki protokołu sięgają roku 1999, kiedy został opracowany przez Jeremiego Millera jako projekt open-source. W 2002 roku grupa deweloperów rozwinęła adaptację protokołu Jabber, która była by odpowiednia jako technologia IETF komunikatorów i statusów.

2. Charakterystyka XMPP

Protokół początkowo został opracowany w środowisku open-source zapewniający otwartą, bezpieczną, wolną od spamu i reklam, rozproszoną alternatywę dla ówczesnych systemów natychmiastowej komunikacji. XMPP oferuje kluczowe zalety w tego typu usługach:

- Otwarty — protokoły XMPP są bezpłatne, otwarte, publiczne i zrozumiałe, ponadto istnieje wiele implementacji w postaci klientów, serwerów, komponentów serwerowych i bibliotek kodu;
- Ustandaryzowany — Internet Engineering Task Force (IETF) sformalizował trzon strumieniowych protokołów XML jako obecną technologię komunikatorów. Specyfikacje zostały opublikowane w dokumencie RFC 3920 i RFC 3921 w 2004 r;
- Sprawdzony — pierwsze technologie Jabber/XMPP zostały opracowane przez Jeremie Miller w 1998 roku i są obecnie dość stabilne; setki programistów pracują nad tymi technologiami, są dziesiątki tysięcy serwerów XMPP uruchomionych w Internecie, a miliony ludzi korzysta z XMPP w komunikatorach za pośrednictwem usług publicznych, np. Google Talk;
- Rozproszony — Architektura sieci XMPP jest podobna do poczty elektronicznej, w wyniku czego, każdy może uruchomić swój własny serwer XMPP, umożliwiając osobom i organizacjom przejąć kontrolę nad doświadczeniami komunikacyjnymi;
- Bezpieczny — każdy serwer XMPP może być odizolowany od sieci publicznej (np. wewnętrzna sieć firmy), solidne bezpieczeństwo zapewniają SASL i TLS wbudowane w specyfikację XMPP, oraz sieć XMPP jest wirtualnie wolna od spamu i reklam;
- Rozszerzalny — przy użyciu XML, każdy może zbudować niestandardowe funkcjonalności. Popularne rozszerzenia są publikowane w serii XEP, taka publikacja nie jest wymagana, a organizacje mogą zachować swoje prywatne rozszerzenia, jeśli jest to pożądane;
- Elastyczny — aplikacje XMPP poza komunikatorami obejmują: zarządzanie siecią, syndykacja, narzędzia do współpracy, udostępnianie plików, gry, zdalne systemy

monitorujące, serwisy webowe, oprogramowanie pośredniczące, chmury obliczeniowe, oraz wiele więcej;

- Różnorodny — szeroka gama firm i projektów open-source używa XMPP do tworzenia i wdrażania usług i aplikacji czasu rzeczywistego, nigdy nie będziesz "zablokowany" po użyciu technologii XMPP.

3. Adresowanie

3.1 Wstęp

System adresowania w protokole XMPP jest zbliżony do adresowania e-mail. Z historycznych powodów adress jednostki XMPP nazywa się Jabber Identifier (JID). Poprawny JID zawiera zbiór uporządkowanych elementów składających się na identyfikator domeny, identyfikator węzła oraz identyfikator zasobu.

Składnia JID jest zdefiniowana przy użyciu notacji ABNF (Augumented Backus-Naur Form) jest następująca:

```
jid = [ node "@" ] domain [ "/" resource ]
domain = fqdn / address-literal
fqdn = (sub-domain 1*("." sub-domain))
address-literal = IPv4address / IPv6address
```

Wszystkie JID-y bazują na powyższej strukturze. W większości wypadków użycie powyższej struktury służy do identyfikacji użytkownika, serwera z którym użytkownik się łączy oraz zasobu połączonego użytkownika w formie <uzytkownik@host/zasob>. Jednak są również inne typy węzłów różnych od klientów są możliwe. Na przykład pewien pokój czatowy oferuje usługę wieloosobowej konwersacji może być zaadresowany jako <pokoj@usluga> (gdzie “pokoj” jest danych pokojem czatowym, a “usluga” jest nazwą hosta tej usługi czatowej).

Każda dozwolona cześć JID (identyfikator węzła, identyfikator domeny oraz identyfikator zasobu) nie może być dłuższa niż 1023 bajty. W efekcie maksymalna długość JID (razem z separatorami “@” oraz “/”) wynosi 3071 bajtów.

3.2 Identyfikator domeny

Identyfikator domenowy jest głównym identyfikatorem i jako jedyny jest wymaganym elementem JID. Zazwyczaj pełni rolę bramy sieciowej lub główny serwer do którego podmioty się podłączają. Jednak podmioty odnosząc się przez identyfikator domenowy nie zawsze muszą być serwerem, ale mogą być usługą zaadresowaną jako poddomena serwera zapewniającego powyższą funkcjonalność i wykraczającymi poza możliwości serwera (wieloosobowa usługa czatowa, katalog użytkowników lub brama do obcego systemu wymiany wiadomości).

Identyfikatorem domenowym dla każdego serwera lub usługi, która komunikuje się przez sieć może być adres IP, ale powinna to być pełna nazwa domenowa (FQDN).

3.3 Identyfikator węzła

Identyfikator węzła jest opcjonalnym drugorzędnym identyfikatorem położony przed identyfikatorem domeny oddzielony znakiem “@”. Zazwyczaj reprezentuje podmiot żądający dostęp przez sieć do serwera (np. klient), chociaż może również reprezentować inne typy podmiotów. Podmiot reprezentowany przed identyfikator węzła jest zaadresowana w ramach danej domeny. W ramach komunikatorów protokołu XMPP ten adres jest zwany “goły JID” i ma postać <wezel@domena>.

3.4 Identyfikator zasobu

Identyfikator zasobu jest opcjonalnym trzeciorzędnym identyfikatorem położonym po identyfikatorze domeny oddzielony znakiem “/”. Identyfikator zasobu może modyfikować któryś z adresów <wezel@domena> lub po prostu <domena>. Zazwyczaj reprezentuje daną sesję, połączenie (np. urządzenie lub położenie), lub obiekt (uczestnik wieloosobowego pokoju czatowego) należący do podmiotu powiązanego z identyfikatorem węzła.

4. XMPP Stanzas

4.1 Wstęp

Dwie podstawowe koncepcje tworzą szybką, asynchroniczną wymianę: strumień XML oraz zwrotka XML. Są one zdefiniowane następująco:

Def. Strumień XML

Strumień XML jest kontenerem dla wymiany elementów XML pomiędzy dwoma punktami sieci. Otwarcie strumienia oznaczamy jednoznacznie poprzez otwarcie znacznika `<stream>` (z odpowiednimi atrybutami i deklaracjami przestrzeni nazw). Koniec strumienia natomiast oznaczamy tagiem `</stream>`. Pomiędzy tymi znacznikami możemy przesyłać nieograniczoną liczbę elementów XML takich jak elementów używanych do negocjacji TLS lub SASL, lub zwrotek XML.

Def. Zwrotka XML

Zwrotka XML jest semantyczną jednostką, która jest przesyłana z jednego punktu do innego przez strumień XML. Zwrotka XML jest bezpośrednim elementem podrzędnym korzenia XML `<stream>`. Początek zwrotki jest oznaczona jednoznacznie przez początkowy tag na głębokości równej 1 strumienia XML (np. `<presence>`). Koniec zwrotki jest jednoznacznie oznaczony towarzyszącym tagiem zamykającym na głębokości równej 1 (np. `</presence>`). Zwrotka XML może zawierać elementy potomne, jeśli to tylko potrzebne w celu przekazania żądanej informacji.

Rozważmy przykładową sesję klienta z serwerem. Aby połączyć się z serwerem klient musi zainicjować strumień XML przesyłając tag otwierający `<stream>` do serwera. Opcjonalnie może przed tym zadeklarować wersję XML oraz kodowanie przed przesłaniem tagu `<stream>`. Serwer powinien odpowiedzieć drugim strumieniem XML do klienta (podobnie jak klient opcjonalnie specyfikując wersję XML oraz kodowanie). Gdy klient dokonał negocjacji SASL, może wysyłać nieograniczoną liczbę zwrotek XML poprzez strumień do dowolnego odbiorcy sieci. Kiedy klient zechce zakończyć strumień przesyła do serwera po prostu tag zamykający `</stream>`. Opcjonalnie serwer może również zakończyć strumień. Po tej operacji klient powinien zakończyć również połączenie TCP. Powyżej opisaną komunikację można zobrazować następująco:

```
|--------------------|
| <stream>           |
|--------------------|
| <presence>         |
|   <show/>          |
| </presence>        |
|--------------------|
| <message to='foo'> |
|   <body/>          |
| </message>         |
|--------------------|
| <iq to='bar'>      |
|   <query/>         |
| </iq>              |
|--------------------|
| ...                |
|--------------------|
| </stream>          |
|--------------------|
```

4.2 Zwrotka <presence/>

Zwrotka <presence/> jest używana do mechanizmu rozgłaszania lub mechanizmu typu “publikuj-subskrybuj” za pomocą którego wiele podmiotów otrzymuje informacje o podmiocie który subskrybują (w tym przypadku informacje o dostępności w sieci). W ogólności podmiot publikujący powininen wysłać zwrotkę `<presence/>` bez ustawionego atrybutu `to`, w przypadku którego serwer do którego jest podłączony powinien rozesłać tę zwrotkę do wszystkich subskrybentów. Jednak publikujący może również ustawić atrybut `to`, wtedy serwer powinien skierować zwrotkę do przeznaczonego adresata.

4.3 Zwrotka <message/>

Zwrotka <message/> jest rodzajem zwrotki przeznaczonym do przekazywania informacji od jednego podmiotu do innego, podobnie do komunikacji jaka występuję w transporcie email. Wszystkie zwrotki wiadomości powinny posiadać atrybut `to`, który określa wybranego adresata wiadomości.

4.4 Zwrotka <iq/>

Zwrotka <iq/> jest mechanizmem typu zapytanie-odpowiedź, zbliżonym w pewnych aspektach do HTTP. Dane zawarte w zapytaniu i odpowiedzi są zdefiniowane w deklaracji przestrzeni nazw bezpośrednio w elemencie potomnym elementu `<iq>` oraz interakcja jest śledzona poprzez atrybut `id`.

5. Serwery XMPP

Serwery obsługujące protokół XMPP zapewniają komunikację pomiędzy klientami, jak i pomiędzy serwerami. Każdy serwer posiada swoje, tzw. drzewo dostarczeń, aby obsłużyć przychodzące zwrotki. Takie drzewo określa, czy dana zwrotka ma być przesłana do innego serwera XMPP, przetworzona lokalnie lub dostarczona do zasobu powiązanego z połączonym węzłem.

Jeśli dana zwrotka nie posiada adresu w polu `to` serwer powinien przetworzyć zwrotkę w imieniu nadawcy. Wszystkie zwrotki muszą posiadać atrybut `to`, więc ta reguła odnosi się tylko do zwrotek otrzymanych od zarejestrowanego podmiotu (np. klient), który jest połączony do serwera. Takie zwrotki serwer powinien rozgłosić do wszystkich podmiotów subskrybujących nadawcę.

Jeśli nazwa hosta identyfikatora domeny z JID zawartego w atrybucie `to` nie jest taka sama jak jedna ze skonfigurowanych nazw hostów na serwerze, lub jego poddomena serwer powinien przekierować zwrotkę do tej obcej domeny. Jeśli serwer ma już otwarty strumień pomiędzy dwoma domenami to serwer przekazuje zwrotkę do autorytatywnego serwera obcej domeny poprzez istniejący strumień. Jeśli nie ma takiego otwartego strumienia, musi rozwiązać nazwę hosta obcej domeny, wynegocjować strumień pomiędzy dwoma domenami, a następnie przekierować zwrotkę do autorytatywnego serwera obcej domeny poprzez nowo utworzony strumień. Jeśli przekierowanie zwrotki się nie powiedzie, serwer musi zwrócić błąd nadawcy.

Jeśli nazwa hosta identyfikatora domeny z JID zawartego w atrybucie `to` pasuje do skonfigurowanej nazwy hosta serwera to musi przetworzyć daną zwrotkę samodzielnie, lub przekierować zwrotkę do usługi, która jest odpowiedzialna za tę poddomenę (jeśli poddomena jest skonfigurowana) lub zwrócić błąd do nadawcy (jeśli poddomena nie jest skonfigurowana).

6. Instalacja ejabberd

6.1 Wstęp

Instalacja serwera jest ogólnie prosta i w większości systemów wystarczy pobrać z gotowych paczek `apt-get install ejabberd`. Po tej operacji możemy przystąpić do konfiguracji serwera. Za pomocą polecenia `ejabberdctl register <uzytkownik> <host> <haslo>` rejestrujemy pierwszego użytkownika. W pliku `/etc/ejabberd/ejabberd.cfg` znajduje się plik konfiguracyjny. Jest napisany w języku Erlang, jednak składnia jest łatwa do zrozumienia. Odnajdujemy sekcję `%% Admin user` i ustawiamy nowo zarejestrowanego użytkownika `{acl, admin, {user, "<uzytkownik>", "<host>"}}`. Dzięki temu uzyskamy dostęp do webowego panelu administracyjnego, który jest dostępny domyślnie na porcie 5280. Restartujemy serwer `service ejabberd restart`. Za pomocą dowolnego klienta XMPP (np. Psi) możemy się połączyć z naszym lokalnym serwerem XMPP.

6.2 Zadania

- Zarejestruj jednego użytkownika user1 za pomocą polecenia ejabberdctl
- Zarejestruj drugiego użytkownika za pomocą klienta XMPP
- Sprawdź działanie serwera przesyłając wiadomość od jednego do innego użytkownika za pomocą klienta XMPP
- Za pomocą narzędzia ejabberdctl poinformuj zalogowanych użytkowników o zatrzymaniu serwera
- Za pomocą narzędzia ejabberdctl wyrzuć sesję wybranego użytkownika

7. Przydatne linki

http://xmpp.org - xmpp standards foundation
http://xmpp.org/xmpp-software/servers - serwery xmpp
http://xmpp.org/xmpp-software/clients - klienci xmpp
http://www.ejabberd.im/files/doc/guide.html - opis instalacji i konfiguracji serwera ejabberd

8. Bibliografia

1. Peter Saint-Andre, Kevin Smith, and Remko Tronçon. XMPP: The Definitive Guide. http://oriolrius.cat/blog/wp-content/uploads/2009/10/Oreilly.XMPP.The.Definitive.Guide.May.2009.pdf
2. RFC 3920. “Extensible Messaging and Presence Protocol (XMPP): Core”. http://xmpp.org/rfcs/rfc3920.html